UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/775,732	02/09/2004	Bhasker Allam	010327-007210US	1552

20350 7590 11/04/2008

TOWNSEND AND TOWNSEND AND CREW, LLP
TWO EMBARCADERO CENTER
EIGHTH FLOOR
SAN FRANCISCO, CA 94111-3834

EXAMINER	
ZHANG, SHIRLEY X	

ART UNIT	PAPER NUMBER
2444	

MAIL DATE	DELIVERY MODE
11/04/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/775,732	ALLAM ET AL.	
	Examiner	Art Unit	
	SHIRLEY X. ZHANG	2444	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *13 August 2008*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1,2,4-13,15 and 16* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1,2,4-13,15 and 16* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Claims 1-2, 4-13, and 15-16 were previously pending;

Claims 1, 2, 7 and 13 have been amended;

Claims 1-2, 4-13, and 15-16 are now pending;

Continued Examination Under 37 CFR 1.114

1. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application on August 13, 2008 after a final rejection was mailed on April 14, 2008.

As this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on April 08, 2008 has been entered.

Response to Amendment

2. Applicant's arguments and amendments filed on August 13, 2008 have been carefully considered. The examiner's full response can be found below in the "Claim Rejections" section.

Regarding the newly added claim limitation "dynamic libraries shared among the plurality of virtual routers", Examiner cites a third reference (Haut, "Summary: Advantages of Shared Libraries") to demonstrate that "dynamic libraries" is a technology well known by one of ordinary skill in the art at the time the present invention was made.

Applicant has amended claims 1, 7 and 13 as an attempt to overcome the prior art rejections made in the "Final Rejection" mailed on April 14, 2008.

However, the new amendments do not place the application in the condition for allowance because they do not sufficiently show that the claimed invention is unobvious to try, or produce unexpected results.

Claim Rejections - 35 USC § 101

35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

3. **Claims 1-2, 4-13 and 15-16** are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

Claims 1, 7 and 13 recite(s) the limitation, "a routing device"

Applicant's specification states the following:

[0031] In an exemplary implementation, the present invention is implemented using software in the form of control logic, in either an integrated or a modular manner.

Therefore, Applicant's specification provides evidence that Applicant intends the "routing device" to include strictly software per se (i.e. "software system", or "computer program"). While the claim is directed towards a device, the device is made up of only software, and as such, the device itself is directed towards software.

Computer programs claimed as computer listings per se, i.e., the descriptions or expressions of the programs are not physical "things". They are neither computer components nor statutory processes, as they are not "acts" being performed. Such claimed computer programs do not define any structural and functional interrelationships between the computer program and other claimed elements of a computer, which permit the computer program's functionality to be realized.

MPEP 2601.1 Section I states, "Since a computer program is merely a set of instructions capable of being executed by a computer, the computer program itself is not a process and USPTO personnel should treat a claim for a computer program, without the computer-readable medium needed to realize the computer program's functionality, as nonstatutory functional descriptive material."

Claims 2, 4-6, 8-12 and 15-16 are dependent on claims 1, 7 or 13, but fail to further limit the respective independent claims to statutory subject matter, therefore they inherit the 35 U.S.C. 101 issue of the independent claim.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. **Claims 1-16** are rejected under 35 U.S.C. 103(a) as being anticipated by IP Infusion ("Virtual Routing for Provider Edge Applications", a white paper by **IP Infusion**, Inc.), in view of Huang et al. ("The ENTRAPID Protocol Development Environment", hereinafter "**Huang**") and Haut et al ("Summary: Advantages of shared libraries", hereinafter "Haut").

Regarding claim 1, IP Infusion teaches a routing device (Fig. 4 discloses a virtual router system) comprising:

An operating system kernel (IP Infusion, page 4, section "Virtual Router Design Considerations" discloses VxWorks as a possible choice of a real-time operating system);

a plurality of virtual routers (Fig. 4 discloses a plurality of virtual routers VR1, VR 2 and VRn that are situated inside the virtual router system), wherein the plurality of virtual routers comprise a plurality of instructions for controlling a data processor to perform one or more tasks, the instructions being stored on a computer readable medium, and wherein the plurality of virtual routers are external to the operating system kernel,

wherein each virtual router further comprises:

a routing protocol stack configured to handle a plurality of routing protocols (IP Infusion, Fig. 3 and page 5, section "Virtual router system overview" disclose a virtual router supporting a plurality of routing protocols including RIP, OSPF and BGP);

a plurality of interface drivers configured to communicate with a plurality of corresponding physical interfaces (IP Infusion, page 6, column 2, paragraph 2 discloses that multiple interfaces can be assigned to a single VR, where it is well known in the art that every physical interface inherently corresponds to an interface driver);

an Internet Protocol (IP) stack configured to interact with the routing protocol stack and perform a forwarding function via the plurality of interface drivers (Fig. 3 discloses that the virtual router contains a TCP/IP stack that interacts with routing protocols such as RIP, OSPF and BGP on one side, and connects to a forwarding plane to perform forwarding function via the physical interfaces on the other side, as is described in page 4, column 1, paragraph 2; The forwarding plane inherently forwards packets to the physical interfaces via interface drivers.), the IP stack having a forwarding information table, information from which is used to perform the forwarding function (page 6, column 1, paragraph 3 discloses that the TCP/IP stack includes many features of a VR, such as software forwarding and management of the FIBs);

a router manager configured to manage the plurality of virtual routers (Fig. 4 discloses a Global Management Authority (GMA) that creates and manages virtual routers); and

an application, wherein the application is situated external to the plurality of virtual routers (The Global Management Authority (GMA) disclosed in Fig. 4 is a software package that performs not only router manager functions, but also other system administrative functions such as remote login, therefore it is interpreted to anticipate said application recited in the claim.),

wherein the application is able to selectively communicate with one or more of the plurality of virtual routers on a dynamic basis to have the one or more virtual routers perform a plurality of tasks (Page 3, column 1 discloses that GMA is created so that global administrators can login to the router system in runtime and then selectively login to a chosen VR to execute commands).

IP Infusion does not expressly teach but Huang teaches a socket layer having a corresponding socket application programming interface, the socket layer configured to facilitate

interactions between the IP stack and the routing protocol stack and the application, wherein the socket application programming interface is used to facilitate communications with the socket layer (Huang, page 4, column 1, paragraph 1 discloses that applications built using the BSD socket API can be ported immediately to a virtualized network kernel (VNK) which is a user-space IP protocol stack. Huang's disclosure implies that VNK supports a BSD socket API.)

It would have been obvious for one of ordinary skill in the art to apply Huang's teaching of socket API to IP Infusion's virtual router system such that a socket layer is included to facilitate interactions between the IP stack, the routing protocol stack and the application, as is recited in the claim.

One would have been motivated to combine as such for the reason that sockets have been widely used in network programming as an inter-process communication mechanism since its introduction in the 1980s. Furthermore, the IP fusion reference states very specifically in the section "Defining Virtual Routing" on page 2 that "a virtual router is an emulation of a physical router at the software layer" and "Essentially many instances of router an protocol code may be running on a single unit," which disclosure implies that the many instances of router must share the same operating system kernel when running on a single unit. IP Infusion, section "Virtual routing Requirements" on page 2 further discloses that a VR must contain its own instance of the applicable routing protocols and be fully independent from another VR. Such requirements imposed by IP fusion on its VR is met by the ENTRAPID model proposed by Huang. Therefore, one of ordinary skill would have been motivated to combine them at the time the present invention was made.

Furthermore, IP Infusion does not explicitly teach "wherein the routing protocol stack and the IP protocol stack are implemented using dynamic libraries shared among the plurality of virtual routers."

However, Huang teaches using software libraries to implement kernel and process virtualization, where a virtualized kernel includes a TCP/IP stack, i.e., the routing protocol stack and the IP protocol stack (Huang, page 4, "A. Kernel Virtualization" and page 5, "B. Process Virtualization").

Meanwhile, Haut teaches the advantages of using shared libraries (i.e., dynamic libraries), wherein the advantages include needing only one copy of the dynamic library in the memory space at a time even if N tasks are running that use the library, therefore reducing the overall memory footprint of the tasks.

One would have been motivated to combine Huang and Haut because both teaches implementing libraries and tasks in a computer system.

It would have been obvious for one of ordinary skill in the art to integrate Haut's teaching about the advantages of dynamic libraries into Huang's implementation of virtualized kernels and processes as such that the virtualized kernels and processes are implemented as dynamic libraries. The combination yields the highly predictable and desirable results of reducing the memory usage of the many virtualized tasks in Huang, and in turn allowing a computer system to support more virtual routers than it would have otherwise been possible had the virtualized tasks been implemented as static libraries.

The combination of Huang and Haut would have also made dynamic libraries an obvious choice when IP Infusion is combined with Huang.

Regarding claim 2, the combination of IP Infusion, Huang and Haut teaches the routing device of claim 1 wherein software is used to implement the router manager (Page 5, column 2, paragraph 2 discloses that a virtual router comprises a full implementation of a physical router at the software level, which includes the router manage such as the "VR management authority" disclosed in Fig. 3).

Regarding claim 4, the combination of IP Infusion, Huang and Haut teaches the routing device of claim 3.

IP Infusion does not explicitly disclose that the IP stack of each of the plurality of virtual routers resides external to the operating system kernel.

However, Huang discloses a virtualized networking system that comprises an operating system kernel and a plurality of virtualized networking kernels (VNK) and processes residing external to the OS kernel (see page 3, column 2). A VNK is the result of extracting the networking portion of the FreeBSD protocol from the kernel and moving it into the user space, i.e. a VNK is an instance of the user-space IP stack (see page 5, column 1, paragraph 2). Huang further discloses that one or more virtualized processes can run on top of a VNK to implement networking protocols such routing protocols above the IP layer, as the purpose of ENTRAPID is to provide a protocol development environment (see page 4, column 1, paragraph 1 and page 8, column 1, section VII). Thus, a VNK and its corresponding virtualized processes together form a virtual router.

Therefore, it would have been obvious for one of ordinary skill in the art to modify IP Infusion's virtual router system with Huang's teaching such that the routing device further comprises an operating system kernel wherein the IP stack of each of the plurality of virtual routers resides external to the operating system kernel. One would have been motivated to combine as such for the desirable advantages that (1) the resulted system allows each copy of the IP stack to work independently; (2) existing user-space applications can be ported immediately to the virtual router (Huang, page 4, column 1), and (3) developers can monitor and modify any aspect of the entire protocol stack without having to make any changes to the kernel (Huang, page 3, column 2, last paragraph).

Regarding claim 5, the combination of IP Infusion, Huang and Haut teaches the routing device of claim 4.

IP Infusion does not teach but Huang further teaches that the operating system kernel includes an associated socket layer, the socket layer having a corresponding socket application programming interface, and the application is able to communicate with the operating system kernel via the associated socket layer using the corresponding socket application programming interface to have the operating system kernel perform one or more of the plurality of tasks (Huang, page 5, column 2, paragraph 1 discloses that ENTRAPID's virtualization approach has been tested on FreeBSD and can be further applied to Windows NT and Solaris. It is known in the art of networking that in FreeBSD, Solaris and Windows, the TCP/IP stack is implemented in the kernel, and is accessible to user space applications via a socket API, i.e., applications external to the operating system kernel communicate with the kernel via a socket layer and socket API).

Therefore, it would have been obvious for one of ordinary skill in the art to modify IP Infusion's virtual router system with Huang's teaching so that the operating system kernel of the virtual router system includes an associated socket layer. The fact that socket API has been used as a standard way of communication between networking applications and kernel since the 1980s would have motivated one of ordinary skill in the art to make such combination at the time the invention was made.

Regarding claim 6, the combination of IP Infusion, Huang and Haut teaches the routing device as recited in claim 1. IP Infusion further teaches an UNIX system incorporating the routing device (page 5, column 1, paragraph 1 discloses that the virtual router system is implemented on Linux, which is a UNIX operating system).

Regarding claim 7, IP Infusion teaches a routing device comprising:

an operating system kernel (page 5, column 1, paragraph 1 discloses that the virtual router is implemented on Linux, VxWorks and OSE, all of which contain an operating system kernel);

a virtual router (Fig. 3 discloses the virtual router 1 in a virtual router system),

wherein the virtual router includes a routing protocol stack and an IP protocol stack (IP Infusion, Fig. 3 and page 5, section "Virtual router system overview" disclose a virtual router supporting a plurality of routing protocols including RIP, OSPF and BGP);

a router manager configured to manage the virtual router (Fig. 4 discloses a Global Management Authority (GMA) that creates and manages virtual routers);

an application residing external to the virtual router (The Global Management Authority (GMA) disclosed in Fig. 4 is a software package that performs not only router manager functions, but also other system administrative functions such as remote login, therefore it is interpreted to anticipate said application recited in the claim); and

a plurality of physical interfaces (Fig. 5 and page 3, column 1, paragraph 1 disclose that the routing device includes multiple physical interfaces);

wherein the application is able to selectively interact with the virtual router and the operating system kernel on a dynamic basis in order to have the virtual router and the operating system kernel perform a plurality of tasks for the application (Page 3, column 1 discloses that GMA is created so that global administrators can login to the router system in runtime and then selectively login to a chosen VR to execute commands).

IP Infusion does not explicitly teach that the virtual router resides external to the operating system kernel.

However, Huang discloses a virtualized networking system that comprises an operating system kernel and a plurality of virtualized networking kernels (VNK) and processes residing external to the OS kernel (see page 3, column 2). A VNK is the result of extracting the networking portion of the FreeBSD protocol from the kernel and moving it into the user space, i.e. a VNK is an instance of the user-space IP stack (see page 5, column 1, paragraph 2). Huang further discloses that one or more virtualized processes can run on top of a VNK to implement networking protocols such routing protocols above the IP layer, as the purpose of ENTRAPID is

to provide a protocol development environment (see page 4, column 1, paragraph 1 and page 8, column 1, section VII). Thus, a VNK and its corresponding virtualized processes together form a virtual router.

It would have been obvious for one of ordinary skill in the art to modify IP Infusion's virtual router system with Huang's teaching so that the routing device further comprises an operating system kernel wherein the virtual routers resides external to the operating system kernel. One would have been motivated to combine as such for the desirable advantages that (1) the resulted system allows each copy of the IP stack to work independently; (2) existing user-space applications can be ported immediately to the virtual router (Huang, page 4, column 1), and (3) developers can monitor and modify any aspect of the entire protocol stack without having to make any changes to the kernel (Huang, page 3, column 2, last paragraph).

Furthermore, the IP fusion reference states very specifically in the section "Defining Virtual Routing" on page 2 that "a virtual router is an emulation of a physical router at the software layer" and "Essentially many instances of router an protocol code may be running on a single unit," which disclosure implies that the many instances of router must share the same operating system kernel when running on a single unit. IP Infusion, section "Virtual routing Requirements" on page 2 further discloses that a VR must contain its own instance of the applicable routing protocols and be fully independent from another VR. Such requirements imposed by IP fusion on its VR is met by the ENTRAPID model proposed by Huang. Therefore, one of ordinary skill would have been motivated to combine them at the time the present invention was made.

IP Infusion does not explicitly teach that the routing protocol stack and the IP protocol stack are implemented using dynamic libraries shared among a plurality of virtual routers.

However, Huang teaches using software libraries to implement kernel and process virtualization, where a virtualized kernel includes a TCP/IP stack, i.e., the routing protocol stack and the IP protocol stack (Huang, page 4, "A. Kernel Virtualization" and page 5, "B. Process Virtualization").

Meanwhile, Haut teaches the advantages of using shared libraries (i.e., dynamic libraries), wherein the advantages include needing only one copy of the dynamic library in the memory space at a time even if N tasks are running that use the library, therefore reducing the overall memory footprint of the tasks.

One would have been motivated to combine Huang and Haut because both teaches implementing libraries and tasks in a computer system.

It would have been obvious for one of ordinary skill in the art to integrate Haut's teaching about the advantages of dynamic libraries into Huang's implementation of virtualized kernels and processes as such that the virtualized kernels and processes are implemented as dynamic libraries. The combination yields the highly predictable and desirable results of reducing the memory usage of the many virtualized tasks in Huang, and in turn allowing a computer system to support more virtual routers than it would have otherwise been possible had the virtualized tasks been implemented as static libraries.

The combination of Huang and Haut would have also made dynamic libraries an obvious choice when IP Infusion is combined with Huang.

Regarding claim 8, the combination of IP Infusion, Huang and Haut teaches the routing device of claim 7. IP Infusion further teaches that software is used to implement the virtual router and the router manager (Page 5, column 2, paragraph 2 discloses that a virtual router comprises a full implementation of a physical router at the software level).

Regarding claim 9, the combination of IP Infusion, Huang and Haut teaches the routing device of claim 7. IP Infusion further teaches that the virtual router includes:

a routing protocol stack configured to handle a plurality of routing protocols (Fig. 3 discloses a virtual router supporting a plurality of routing protocols including RIP, OSPF and BGP);

a plurality of interface drivers configured to communicate with corresponding physical interfaces (Page 6, column 2, paragraph 2 discloses that multiple interfaces can be assigned to a single VR, where it is well known in the art that every physical interface inherently corresponds to an interface driver);

an Internet Protocol (IP) stack configured to interact with the routing protocol stack and perform a forwarding function via the plurality of interface drivers (Fig. 3 discloses that the virtual router contains a TCP/IP stack that interacts with routing protocols such as RIP, OSPF and BGP on one side, and connects to a forwarding plane to perform forwarding function via the physical interfaces on the other side, as is described in page 4, column 1, paragraph 2; The forwarding plane inherently forwards packets to the physical interfaces via interface drivers.), the IP stack having a forwarding information table, information from which is used to

perform the forwarding function (page 6, column 1, paragraph 3 discloses that the TCP/IP stack includes many features of a VR, such as software forwarding and management of the FIBs); and

IP Infusion does not teach but Huang teaches a socket layer having a corresponding socket application programming interface, the socket layer configured to facilitate interactions between the IP stack and the routing protocol stack and the application, wherein the socket application programming interface is used to facilitate communications with the socket layer (Huang, page 4, column 1, paragraph 1 discloses that applications built using the BSD socket API can be ported immediately to a virtualized network kernel (VNK) which is a user-space IP protocol stack. Huang's disclosure implies that VNK supports a BSD socket API.)

It would have been obvious for one of ordinary skill in the art to apply Huang's teaching of socket API to IP Infusion's virtual router system such that a socket layer is included to facilitate interactions between the IP stack, the routing protocol stack and the application, as is recited in the claim. One would have been motivated to combine as such for the reason that sockets have been widely used in network programming as an inter-process communication mechanism since its introduction in the 1980s.

Regarding claim 10, the combination of IP Infusion, Huang and Haut teaches the routing device of claim 9.

IP Infusion does not teach that the IP stack of the virtual router resides external to the operating system kernel.

However, as mentioned above in claim 7, Huang discloses a virtualized networking system that comprises an operating system kernel and a plurality of virtual routers residing

external to the OS kernel (see page 3, column 2). Each virtual router comprises one VNK and a plurality of corresponding virtualized processes, where the VNK is a user-space IP stack.

It would have been obvious for one of ordinary skill in the art to modify IP Infusion's virtual router system with Huang's teaching such that the IP stack of the virtual router resides external to the operating system kernel. One would have been motivated to combine as such for the desirable advantages that (1) the resulted system allows each copy of the IP stack to work independently; 2) existing user-space applications can be ported immediately to the virtual router (Huang, page 4, column 1), and (3) developers can monitor and modify any aspect of the entire protocol stack without having to make any changes to the kernel (Huang, page 3, column 2, last paragraph).

Regarding claim 11, the combination of IP Infusion, Huang and Haut teaches the routing device of claim 7.

IP Infusion does not teach but Huang further teaches that the operating system kernel includes an associated socket layer, the socket layer having a corresponding socket application programming interface, and the application is able to communicate with the operating system kernel via the associated socket layer using the corresponding socket application programming interface to have the operating system kernel perform one or more of the plurality of tasks (Huang, page 5, column 2, paragraph 1 discloses that ENTRAPID's virtualization approach has been tested on FreeBSD and can be further applied to Windows NT and Solaris. It is known in the art of networking that in FreeBSD, Solaris and Windows, the TCP/IP stack is implemented in

the kernel, and is accessible to user space applications via a socket API, i.e., applications external to the operating system kernel communicate with the kernel via a socket layer and a socket API).

Therefore, it would have been obvious for one of ordinary skill in the art to modify IP Infusion's virtual router system with Huang's teaching so that the operating system kernel of the virtual router system includes an associated socket layer. The fact that socket API has been used as a standard way of communication between networking applications and kernel since the 1980s would have motivated one of ordinary skill in the art to make such combination at the time the invention was made.

Regarding claim 12, the combination of IP Infusion, Huang and Haut teaches the routing device as recited in claim 7. IP Infusion further teaches an UNIX system incorporating the routing device (page 5, column 1, paragraph 1 discloses that the virtual router system is implemented on Linux, which is a UNIX operating system).

Regarding claim 13, IP Infusion teaches a routing device comprising

a plurality of virtual routers (Fig. 4 discloses a plurality of virtual routers VR1, VR 2 and VRn that are situated inside the virtual router system),

wherein each virtual router includes a routing protocol stack (IP Infusion, Fig. 3 and page 5, section "Virtual router system overview" disclose a virtual router supporting a plurality of routing protocols including RIP, OSPF and BGP) and

an application residing external to the plurality of virtual routers (The Global Management Authority (GMA) disclosed in Fig. 4 is also an application situated external to the

plurality of virtual routers; As the instant invention does not disclose the functions of said application, IP Infusion's GMA is interpreted to anticipate said application recited in the claim);

wherein the application is able to selectively interact with one of the plurality of virtual routers (Page 3, column 1 discloses that GMA is created so that global administrators can login to the router system in runtime and then selectively login to a chosen VR to execute commands).

wherein the application is able to selectively communicate with one or more of the plurality of virtual routers on a dynamic basis to have the one or more virtual routers perform a plurality of tasks (Page 3, column 1 discloses that GMA is created so that global administrators can login to the router system in runtime and then selectively login to a chosen VR to execute commands).

IP Infusion does not explicitly disclose that each virtual router has an associated socket layer and an Internet Protocol (IP) stack, wherein the associated socket layer has a corresponding socket application programming interface configured to facilitate communications with the associated socket layer, and the associated socket layer is configured to facilitate interactions between the IP stack and the application

However, Huang discloses in page 4, column 1, paragraph 1 that applications built using the BSD socket API can be ported immediately to a virtualized network kernel (VNK) which is a user-space IP stack. Huang's disclosure implies that the VNK supports a BSD socket API, through which user space applications can communicate with the socket layer and the IP stack.

It would have been obvious for one of ordinary skill in the art to apply Huang's teaching of socket API to IP Infusion's virtual router system such that a socket layer and API is included

to facilitate interactions between the IP stack and the application, as is recited in the claim. One would have been motivated to combine as such because sockets have been widely used in network programming as an inter-process communication mechanism since its introduction in the 1980s.

Furthermore, the IP fusion reference states very specifically in the section "Defining Virtual Routing" on page 2 that "a virtual router is an emulation of a physical router at the software layer" and "Essentially many instances of router an protocol code may be running on a single unit," which disclosure implies that the many instances of router must share the same operating system kernel when running on a single unit. IP Infusion, section "Virtual routing Requirements" on page 2 further discloses that a VR must contain its own instance of the applicable routing protocols and be fully independent from another VR. Such requirements imposed by IP fusion on its VR is met by the ENTRAPID model proposed by Huang. Therefore, one of ordinary skill would have been motivated to combine them at the time the present invention was made.

IP Infusion does not explicitly teach that the routing protocol stack and the IP protocol stack are implemented using dynamic libraries shared among a plurality of virtual routers.

However, Huang teaches using software libraries to implement kernel and process virtualization, where a virtualized kernel includes a TCP/IP stack, i.e., the routing protocol stack and the IP protocol stack (Huang, page 4, "A. Kernel Virtualization" and page 5, "B. Process Virtualization").

Meanwhile, Haut teaches the advantages of using shared libraries (i.e., dynamic libraries), wherein the advantages include needing only one copy of the dynamic library in the memory space at a time even if N tasks are running that use the library, therefore reducing the overall memory footprint of the tasks.

One would have been motivated to combine Huang and Haut because both teaches implementing libraries and tasks in a computer system.

It would have been obvious for one of ordinary skill in the art to integrate Haut's teaching about the advantages of dynamic libraries into Huang's implementation of virtualized kernels and processes as such that the virtualized kernels and processes are implemented as dynamic libraries. The combination yields the highly predictable and desirable results of reducing the memory usage of the many virtualized tasks in Huang, and in turn allowing a computer system to support more virtual routers than it would have otherwise been possible had the virtualized tasks been implemented as static libraries.

The combination of Huang and Haut would have also made dynamic libraries an obvious choice when IP Infusion is combined with Huang.

Regarding claim 15, the combination of IP Infusion, Huang and Haut teaches the routing device of claim 13.

IP Infusion does not teach that the plurality of virtual routers reside external to the operating system kernel.

However, Huang discloses in page 3, column 2 a virtualized networking system that comprises an operating system kernel, and a plurality of virtualized networking kernels (VNK)

and virtualized processes that reside external to the OS kernel. As is further disclosed in page 5, column 1, paragraph 2, a VNK is the result of extracting the networking portion of the FreeBSD protocol from the kernel and moving it into the user space, therefore, a VNK is an instance of the user-space IP stack.

Huang further discloses that one or more virtualized processes can run on top of a VNK to implement networking protocols such routing protocols above the IP layer (see page 4, column 1, paragraph 1 and page 8, column 1, section VII). Thus, a VNK and its corresponding virtualized processes together form a virtual router.

It would have been obvious for one of ordinary skill in the art to modify IP Infusion's virtual router system with Huang's teaching such that the IP stack of the virtual router resides external to the operating system kernel. One would have been motivated to combine as such for the desirable advantages that (1) the resulted system allows each copy of the IP stack to work independently; (2) existing user-space applications can be ported immediately to the virtual router (Huang, page 4, column 1), and (3) developers can monitor and modify any aspect of the entire protocol stack without having to make any changes to the kernel (Huang, page 3, column 2, last paragraph).

Regarding claim 16, the combination of IP Infusion, Huang and Haut teaches the routing device as recited in claim 13. IP Infusion further teaches an UNIX system incorporating the routing device (page 5, column 1, paragraph 1 discloses that the virtual router system is implemented on Linux, which is a UNIX operating system).

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to SHIRLEY X. ZHANG whose telephone number is (571)270-5012. The examiner can normally be reached on Monday through Friday 7:30am - 5:00pm EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, William Vaughn can be reached on (571) 272-3922. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/S. X. Z./
Examiner, Art Unit 2444
10/23/2008
/William C. Vaughn, Jr./

Supervisory Patent Examiner, Art Unit 2444